(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
17 October 2002 (17.10.2002)

PCT

(10) International Publication Number
WO 02/082315 A1

(51) International Patent Classification⁷: G06F 17/30, H04L 12/66, 29/06

(21) International Application Number: PCT/FI02/00280

(22) International Filing Date: 2 April 2002 (02.04.2002)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
20010690 3 April 2001 (03.04.2001) FI

(71) Applicant *(for all designated States except US)*: SONERA OYJ [FI/FI]; Teollisuuskatu 15, FIN-00510 Helsinki (FI).

(72) Inventor; and

(75) Inventor/Applicant *(for US only)*: KOSKELA, Markku [FI/FI]; Perkkaankuja 3 B 44, FIN-02600 Espoo (FI).

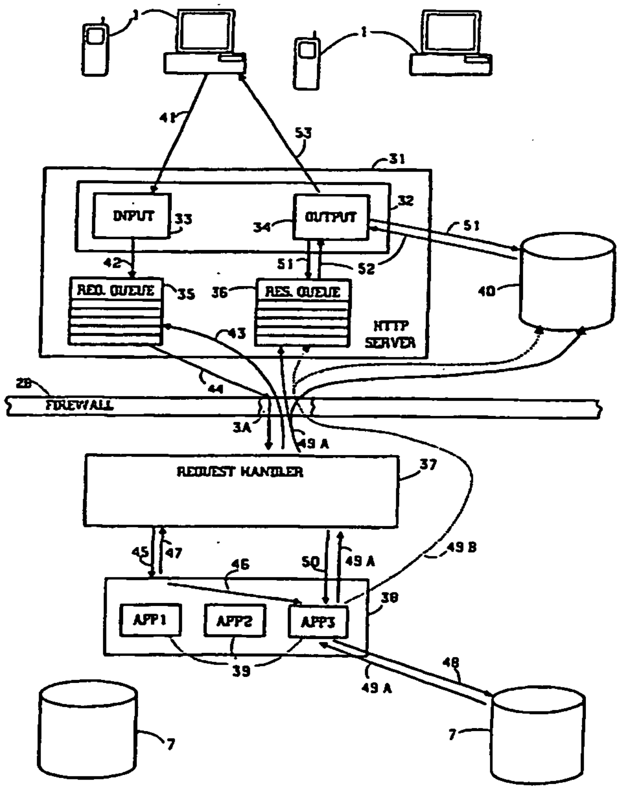
(74) Agent: TAMPEREEN PATENTTITOIMISTO OY; Hermiankatu 6, FIN-33720 Tampere (FI).

(81) Designated States *(national)*: AE, AG, AL, AM, AT, AT (utility model), AU, AZ, BA, BB, BG, BR, BY, BZ, CA, CH, CN, CO, CR, CU, CZ, CZ (utility model), DE, DE (utility model), DK, DK (utility model), DM, DZ, EC, EE, EE (utility model), ES, FI, FI (utility model), GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MA, MD, MG, MK, MN, MW, MX, MZ, NO, NZ, OM, PH, PL, PT, RO, RU, SD, SE, SG, SI, SK, SK (utility model), SL, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VN, YU, ZA, ZM, ZW.

(84) Designated States *(regional)*: ARIPO patent (GH, GM, KE, LS, MW, MZ, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE, TR), OAPI patent

[Continued on next page]

(54) Title: ARRANGEMENT FOR PROCESSING CLIENT REQUESTS



(57) Abstract: This invention relates generally to processing re-quests coming from a client to a network server. In the arrangement according to the invention an HTTP request coming from a client's terminal is received in an HTTP server from where it is picked up by a request handler. The HTTP server contains an application for receiving the client requests and sending responses, a queue for the received client requests, and another queue for the responses. The HTTP server is situated outside the firewall, and the request handler inside the firewall, as are back-end systems for the request handler as well. Since the request handler requires the HTTP server to return a client request in the re-quest queue as a response to the handler a connec-tion through the firewall is opened inside the firewall.

WO 02/082315 A1

(BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

Published:

— *with international search report*

— *before the expiration of the time limit for amending the claims and to be republished in the event of receipt of amendments*

For two-letter codes and other abbreviations, refer to the "Guidance Notes on Codes and Abbreviations" appearing at the beginning of each regular issue of the PCT Gazette.

Arrangement for processing client requests

Field of the Invention

This invention relates generally to processing requests coming from a client to a network server. More particularly, the invention relates to HTTP requests. Furthermore, the invention relates to a security arrangement for services.

Background of the Invention

At present, server arrangements providing services usually contain a firewall, as in FIGS. **1** and **2**. The firewall is a security element whose purpose is to keep unwanted service requests out of the service providing systems. There are two types of firewalls: filtering firewalls that block selected network packets, and proxy servers that make network connections for you. Anyway, it is convenient to think about firewalls as packet filters. Data is only allowed to come to the system if the firewall rules allow it. As packets arrive they are filtered, for example, by their type, source address, destination address, and port information contained in each packet.

FIG. **1** shows an example of a possible arrangement at present. Clients' terminals **1** send service requests **8**, such as HTTP requests, through a firewall **2** to a HTTP server **4**. The requests are directed to the right applications **5**, such as CGI (Common Gateway Interface), API (Application Programming Interface), ISAPI (Internet Server Application Programming Interface), or Java Servlet, which handle the forward processing of each request. The application can use middleware services **6** for processing the request sent forward **9** in a back-end system, i.e. in the system behind the HTTP server. The middleware processing element **6** can use a database **7** for asking **10** necessary data needed for establishing the service requested. The database returns the request **11** as a response back to the processing element or directly to the application. Alternatively, the application can have a direct connection to a relevant database. The applications **5** send **12** responses to the clients' terminals through the firewall.

All connections between the clients' terminals and the HTTP server go through an allowed "hole" **3** in the firewall. The hole lets the packets which are not filtered away go through. However, in situations as in FIG. **1** the firewall has to allow HTTP traffic to go through due to the firewall being situated between the clients' terminals (in the Internet) and the HTTP server. Since the HTTP traffic amount is very huge the firewall cannot establish a very efficient security effect.

Another problem which appears in the system of FIG. **1** is that the connection to the applications and the back-end system are opened (established) outside the service providing arrangement. This exposes the arrangement to thousands of simultaneous HTTP requests, which can create an overload situation in the service arrangement, and even crash the arrangement. Naturally, service providers do not want this to happen.

FIG. **2** shows another example of a possible arrangement at present. Clients' terminals **1** send service requests **8**, such as HTTP requests to a HTTP server **4**. The requests are directed to the right applications **5**, which handle the processing of each request forward. The application can use middleware services **6** for processing the request sent forward **9** in a back-end system, i.e. in the system behind the HTTP server. The requests **9** sent to the back-end system go through a firewall **2A**. The middleware processing element **6** can use a database **7** for asking **10** necessary data needed for establishing the service requested. The database responds **11** back to the processing element or directly to the application through the firewall. Alternatively, the application can have a direct connection to a relevant database. The applications **5** send **12** responses to the clients' terminals.

All connections from the clients' terminals go straight to the HTTP server, which directs them to the relevant application. As can be noticed in FIG. **2** the firewall has to allow different traffic types to go through due to the different applications. So the firewall has to have several "holes" **3** for letting the packets, which are not filtered away, go through. As a result of having several "holes" in the firewall **2A**, this solution is also exposed to security risks.

Also in FIG. **2** the connections to the applications and the back-end system are opened (established) outside the service providing arrangement, exposing the arrangement to thousands of simultaneous HTTP requests, which can create an overload situation in the service arrangement, and even crash the arrangement.

U.S. patent application 6,141,759 also shows a present solution wherein connections are opened (established) outside the firewall making it possible to crash the system of a service provider. The intention of the invention is to increase the security level of a service providing arrangement and eliminate the possibility of crashing the arrangement from the outside.

Summary of the Invention

The idea of the invention is that a HTTP request coming from a client's terminal is picked up by a request handler from a HTTP server. The HTTP server contains an application for receiving the client requests and sending responses, a queue for the received client requests, and another queue for the responses. The HTTP server is situated outside the firewall, and the request handler inside the firewall, as are back-end systems for the request handler as well. Since the request handler requires the HTTP server to return a client request in the request queue as a response to the handler a connection through the firewall is opened, i.e. established, inside the firewall. In other words, the request handler in the firewall controls the traffic through the firewall. This arrangement eliminates situations where HTTP requests coming from the Internet overload the service providing systems.

The request handler sends the requests forward to the back-end systems, wherein the requests are handled for establishing responses for sending them back to the clients. The responses are sent through the firewall either to the response queue or to a special database from where the HTTP server can pick them up.

Since all connections through the firewall are opened inside the firewall the security of the service providing arrangement is more reliable than in the present solutions.

Brief Description of the Drawings

In the following the invention is described in more detail by means of FIGS. **1** - **5** in the attached drawings where

FIG. **1** illustrates an example of a present solution for processing HTTP requests,

FIG. **2** illustrates another example of a present solution for processing HTTP requests,

FIG. **3** illustrates an example of an arrangement according to the invention,

FIG. **4** illustrates an example of a flow chart describing the method according to the invention, and

FIG. **5** illustrates an example of another arrangement according to the invention.

Detailed Description of the Invention

FIG. **3** shows an example of an arrangement according to the invention. A client terminal **1** sends an HTTP request **41** to the HTTP server **31**. The HTTP server contains an application **32** which handles the receiving of HTTP requests in an input processing element **33**, and sending of responses back to clients' terminals in an output processing element **34**. The received HTTP requests are forwarded **42** for stocking them in a request queue **35**. The HTTP server also contains a response queue **36** for responses to the clients' terminals. The HTTP server is located outside a firewall **2B**.

Inside the firewall there is an element **37**, called request handler, which handles the creation of connections through the firewall. The request handler also directs the HTTP request to a relevant application **39** for establishing the service requested.

The request handler is preferably middleware software. The definition of middleware is not accurate, but usually middleware is considered to be a layer or software between the network and the applications. Middleware makes advanced network applications much easier to use. Possible middleware techniques for creating the request handler are, for example, CORBA, TUXEDO, COM, DCOM, RPC and RMI.

The request handler **37** inquires **43** from the request queue **35** in the HTTP server **31** if a request is available in the queue, requiring a response **44** to the request handler. At the same time when sending **43** an inquiry, the request handler creates a connection through the firewall, i.e. open a "hole" **3A** in the firewall. If there is a request in the request queue, it can be put into a response for the request of the request handler and sent it to the request handler through the hole of the firewall **2B**.

The request handler inquires **45** from an application logic **38**, which application **39** is the right one for the request. The application logic maps **46** (using for example URL information) the application and the HTTP-request, and returns **47** the mapping information to the request handler. The request handler sends **50** the HTTP request to the right application **39**. It can be possible that the application logic is combined with the request handler, but keeping them separate is preferable.

As can be noticed, the request handler acts like a client process, which uses outside services, i.e. the HTTP server, the application logic, and the applications.

If needed, the application **39** can use a database **7** for querying **48** the data needed for establishing the service request. The response data is delivered back **49A** through the application and the request handler to the response queue through the firewall or to a special database **40** outside the firewall. The special database is used if the response contains a great amount of data wherefore it is inefficient or impossible to use the response queue. Alternatively, the response data is delivered back **49B** just through the application to the response queue or the special database.

The output processing element **34** asks **51** the response queue **36** or the special database **40** responses ready for delivering to the clients' terminals **1**. If there are responses in the queue or in the database the responses are conducted **52** to the output processing element **34**, which delivers **53** them to the clients terminals.

FIG. **4** shows an example of a flow chart describing the method according to the invention. First, the input processing element **33** in the HTTP server receives **60** a HTTP request from a client's terminal. The received HTTP request is stocked **61** in the request queue. The request handler, which is on the other side of the firewall, inquires **62** received HTTP requests in the request queue. Due to this the request handler opens a connection through the firewall - from the safe side of the firewall. As a response to the inquiry the received HTTP request is returned **63** to the request handler through the firewall.

Next, the request handler inquires **64** about a relevant application for handling the HTTP request from the application logic. As a response to this inquiry the application logic maps **65** the relevant application and the HTTP request together, and returns **66** the mapping information to the request handler. The request handler sends **67** the HTTP request to the relevant application.

The application can ask **68** necessary data, if needed, for a request response from a database. If the data from the database is needed for performing the request response, the response from the database is conducted **69** to the application. Alternatively, the application can form the request response without using the database.

The application sends **70** the request response either direct to the response queue in the HTTP server or to the special database on the other (unsafe) side of the firewall, or through the request handler to the response queue in the HTTP server or to the special database on the other side of the firewall.

The output processing element in the HTTP server inquires **71** about request responses from the response queue and the special database. If a request response exists the request response in the response queue or in the special database the output processing element delivers (sends) **72** it to the client's terminal.

The arrangement according to the invention offers a very robust environment for providing services. The arrangement is almost linearly scalable. The request handler can pick up HTTP requests from several HTTP servers and queues as depicted in FIG. **5**. On the other hand, there can also be several request handlers, which are capable of delivering requests to the same applications. The arrangement is stable as well since the HTTP servers and the request handlers can be cross-connected in a way that the request handlers can pick up a HTTP request from the queue of the same HTTP server.

The processing of HTTP requests can be prioritized. HTTP servers can contain several request and response queues, which can be used for the prioritization. This means that the HTTP server places HTTP request into different queues according to certain criteria. The criteria can be, for example, the URL requested or a part of it, session ID, client's IP address, or client's phone number. Each queue can be connected to a different request handler.

Request handlers can vary from each other. For example, certain request handlers are optimized for fast handling, others for taking into account security needs, and some request handlers for handling a certain type of traffic, such as management traffic or high priority services. Request handlers can also provide authentication and authorization tasks, and also session management. Request handlers can also support transaction management.

As can be noticed, the arrangement according to the invention can be realized in many ways. The application handling input/output processing in the HTTP server can be performed by using a common application interface technique, such as CGI, NSAPI, ISAPI, or JavaServlet. Request and response queues can act in a FIFO (First In First Out) principle. The queues provide read (GetRequest) and write (AddRequest) actions. The services of the queues can be performed using different techniques, such as middleware (CORBA, TUXEDO, DCOM, COM, RPC, RMI). Middleware techniques can be used when performing the request handler as mentioned before.

The application logic can also be performed using middleware techniques. The application logic can be thought to be a service, from which the request handler can ask the relevant application for the HTTP request under processing. The special database can also be seen as a service, through which applications can deliver huge responses to the HTTP server, past the response queue. There are several ways for providing the database: using a normal file system with, for example, FTP or NFS, using some database technique, or modeling the database as a service, such as CORBA.

As can be noticed, the request handler preferably acts as a client that uses outside services, but this is not the only solution for performing an arrangement according to the invention. Although, the invention is described in this text handling HTTP request from clients' terminals, such as a Web browser or WAP mobile phone it should be mentioned that it is possible to handle other kinds of requests as well. According to the matters mentioned above, it is clear that the arrangement according to the invention can be performed in many ways, in the scope of the inventive idea.

Claims

1. An arrangement for providing services in a communication network environment, which comprises at least one client's terminal, one server for connecting the services and the clients' terminals, one back-end system for each server, and a firewall between the server and the back-end system, **characterized** in that the server comprises

- an application for receiving service requests from the clients' terminals, and for sending responses to the clients' terminals,
- a request queue for the received service requests, and
- a response queue for responses to be sent to the clients terminals,

and the back-end system, which handles the performing of the responses, comprises a request handler to open a connection through the firewall for picking up the requests from the request queue.

2. An arrangement according to claim 1, **characterized** in that the arrangement further comprises a special database for keeping the responses, which are too large for the response queue to handle.

3. An arrangement according to claim 2, **characterized** in that the application for receiving the service requests further comprises

- an input processing element for receiving the service request and delivering them to the request queue and
- an output processing element for inquiring for the responses from the response queue and from the special database and for sending the responses to the clients' terminals.

4. An arrangement according to claim 3, **characterized** in that the back-end system further comprises an application logic for mapping together the requests from the request handler and applications that handle the requests.

5. An arrangement according to claim 3 or 4, **characterized** in that the back-end system further comprises at least one database from where the applications, which handles the requests, can ask for necessary data for the responses which are sent to the response queue or to the special database.

6. An arrangement according to claim 5, **characterized** in that the special database is performed to be a service for the other elements of the arrangement which use the special database.

7. An arrangement according to claim 6, **characterized** in that the request and the response queue are performed to be services for the other elements of the arrangement which use them.

8. An arrangement according to claim 7, **characterized** in that the application logic is performed to be a service for the other elements of the arrangement which use it.

9. An arrangement according to claim 8, **characterized** in that the request handler is performed to be a client which uses the services in the arrangement.

10. An arrangement according to claim 1–9, **characterized** in that the service requests are HTTP requests.

11. A method for providing services in a communication network environment, which comprises at least one client's terminal, one server for connecting the services and the clients' terminals, one back-end system for each server, and a firewall between the server and the back-end system, the method comprising the steps of receiving a service request from the clients terminal in the server and sending a response from the server to the clients terminal, **characterized** in that the method further comprises the steps of:

- stocking the received service request in a request queue in the server,

- opening a connection from the back-end side of the firewall for inquiring about a request in the request queue by a request handler,
- returning the request from the request queue in the server side of the firewall to the request handler in the back-end side of the firewall, as a response to the inquiry, and
- sending the response from the back-end side of the firewall to a predetermined element in the server side of the firewall.

12. A method according to claim 11, **characterized** in that the method further comprises the steps of:

- requesting a relevant application from an application logic in the back-end side of the firewall by the request handler and
- sending the request from the request handler to the relevant application for forming the response.

13. A method according to claim 12, **characterized** in that between the steps of inquiring about the relevant application and sending the request from the request handler to the relevant application the method further comprises the steps of:

- mapping the relevant application and the request in the application logic and
- sending the mapping information from the application logic to the request handler

14. A method according to claim 12 or 13, **characterized** in that the method further comprises, if needed, the step of asking for necessary data from a database and the step of returning the necessary data as a response to the relevant application for forming the response.

15. A method according to claim 12, 13, or 14, **characterized** in that the response is sent from the application to the predetermined element, which is a response queue in the server.

16. A method according to claim 15, **characterized** in that the response is sent through the request handler.

17. A method according to claim 12, 13, or 14, **characterized** in that the response is sent from the application to the predetermined element, which is a special database in the server side of the firewall.

18. A method according to claim 17, **characterized** in that the response is sent through the request handler.

19. A method according to claim 11 or 16, **characterized** in that before the step of sending the response from the server to the client's terminal the method further comprises the step of inquiring for the response form the response queue by an output processing element.

20. A method according to claim 17 or 18, **characterized** in that before the step of sending the response from the server to the client's terminal the method further comprises the step of inquiring for the response form the special database by an output processing element.

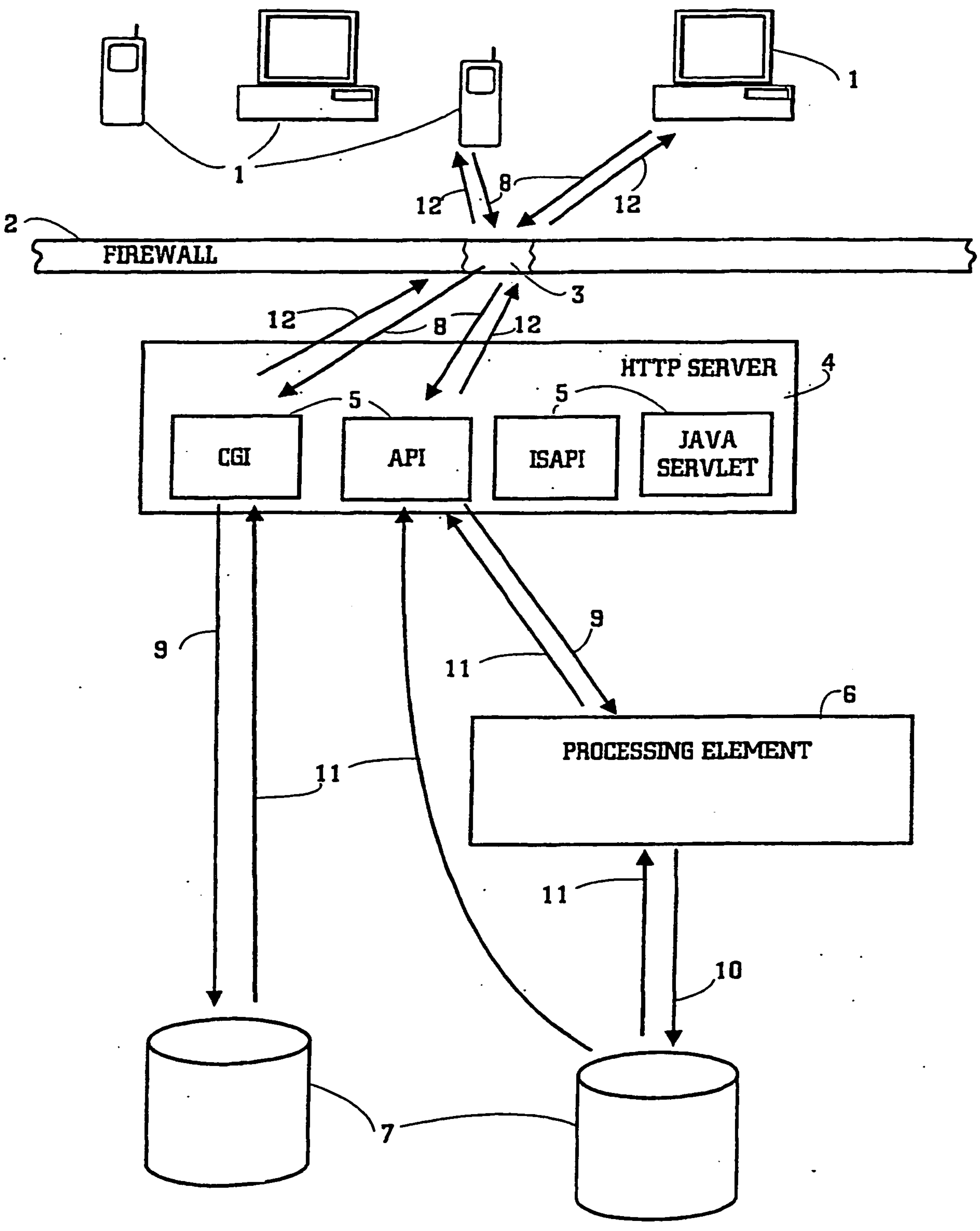

1
1
2
FIREWALL
3
8
12
12
8
12
12
HTTP SERVER
4
5
5
CGI
API
ISAPI
JAVA SERVLET
9
9
11
11
11
6
PROCESSING ELEMENT
11
10
7

FIG. 1

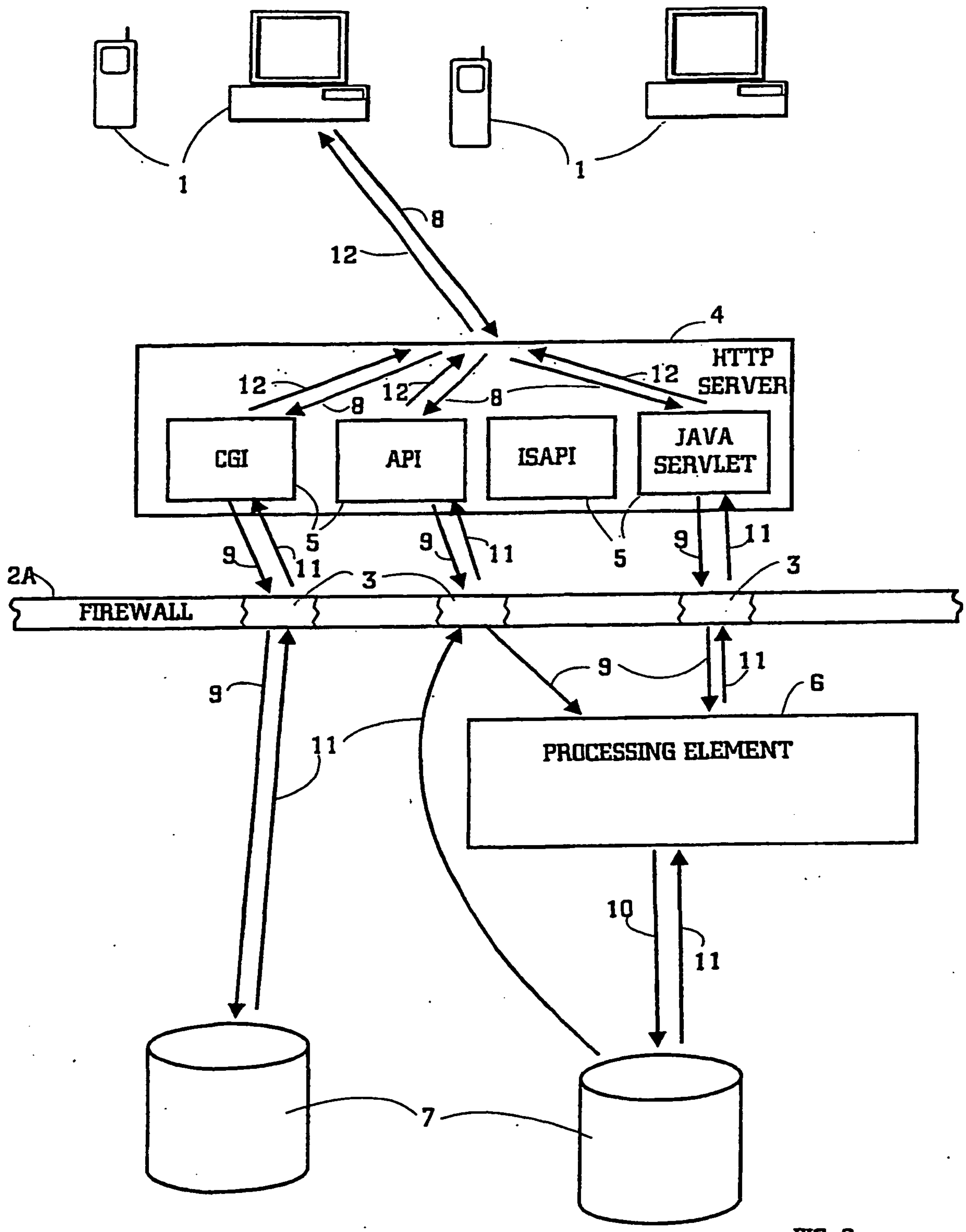

1
1
8
12
4
HTTP SERVER
12
8
12
8
12
CGI
API
ISAPI
JAVA SERVLET
5
5
5
9
11
9
11
9
11
2A
3
3
3
FIREWALL
9
11
6
9
11
PROCESSING ELEMENT
10
11
7

FIG. 2

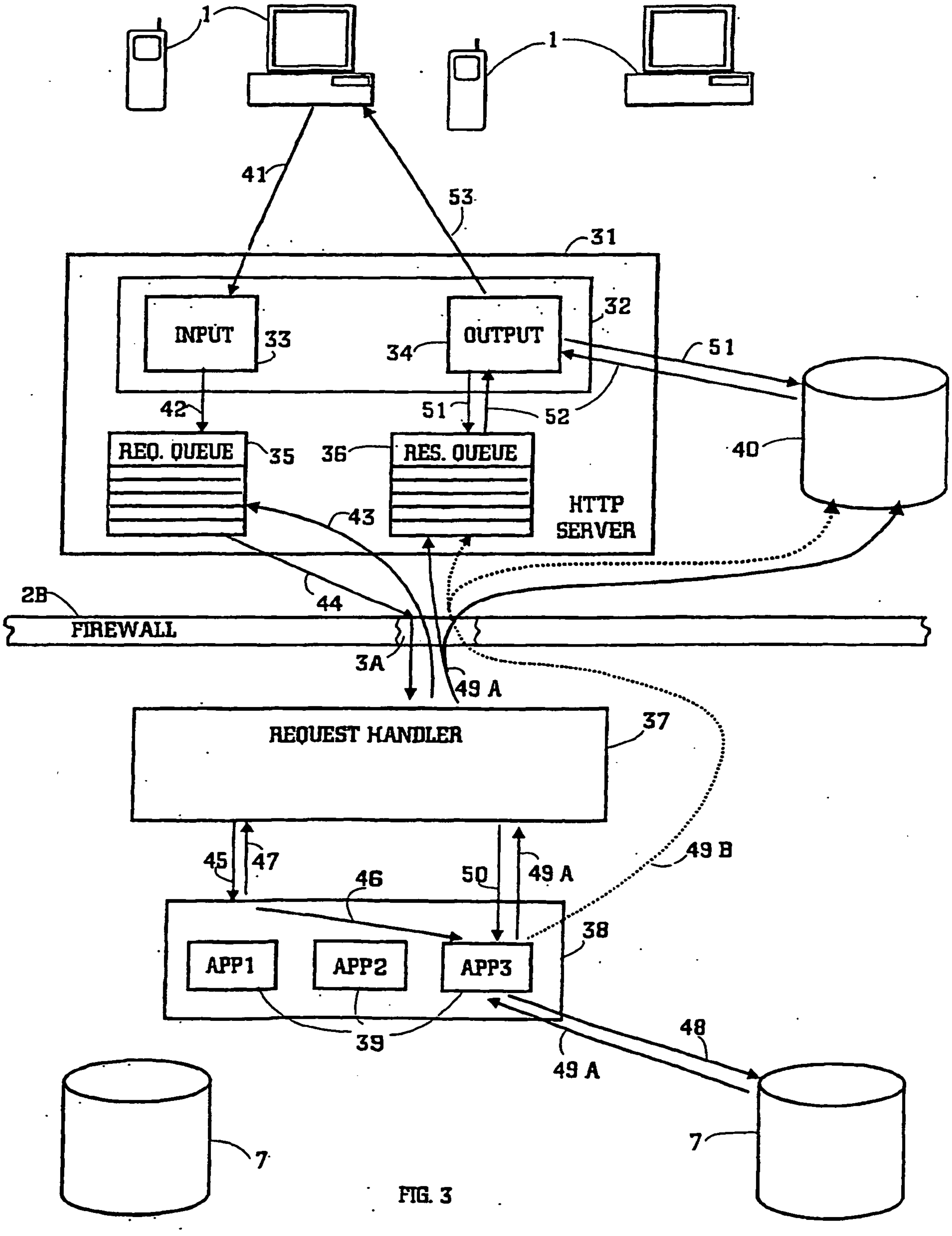

1
1
41
53
31
INPUT
33
OUTPUT
32
34
51
42
51
52
REQ. QUEUE
35
36
RES. QUEUE
40
43
HTTP SERVER
2B
44
FIREWALL
3A
49 A
REQUEST HANDLER
37
45
47
46
50
49 A
49 B
38
APP1
APP2
APP3
39
48
49 A
7
7

FIG. 3

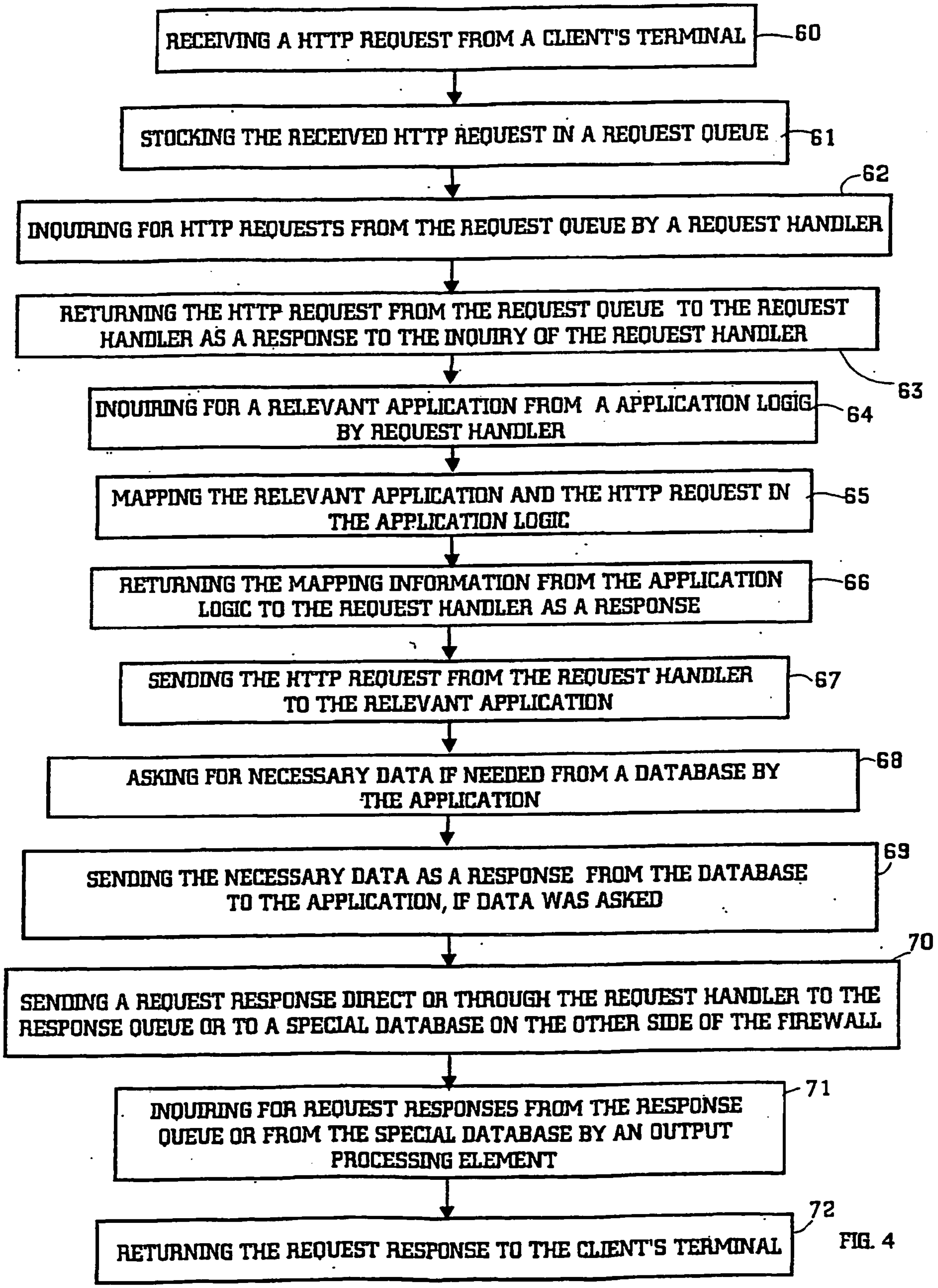

RECEIVING A HTTP REQUEST FROM A CLIENT'S TERMINAL
60
STOCKING THE RECEIVED HTTP REQUEST IN A REQUEST QUEUE
61
62
INQUIRING FOR HTTP REQUESTS FROM THE REQUEST QUEUE BY A REQUEST HANDLER
RETURNING THE HTTP REQUEST FROM THE REQUEST QUEUE TO THE REQUEST HANDLER AS A RESPONSE TO THE INQUIRY OF THE REQUEST HANDLER
63
INQUIRING FOR A RELEVANT APPLICATION FROM A APPLICATION LOGIC BY REQUEST HANDLER
64
MAPPING THE RELEVANT APPLICATION AND THE HTTP REQUEST IN THE APPLICATION LOGIC
65
RETURNING THE MAPPING INFORMATION FROM THE APPLICATION LOGIC TO THE REQUEST HANDLER AS A RESPONSE
66
SENDING THE HTTP REQUEST FROM THE REQUEST HANDLER TO THE RELEVANT APPLICATION
67
ASKING FOR NECESSARY DATA IF NEEDED FROM A DATABASE BY THE APPLICATION
68
SENDING THE NECESSARY DATA AS A RESPONSE FROM THE DATABASE TO THE APPLICATION, IF DATA WAS ASKED
69
70
SENDING A REQUEST RESPONSE DIRECT OR THROUGH THE REQUEST HANDLER TO THE RESPONSE QUEUE OR TO A SPECIAL DATABASE ON THE OTHER SIDE OF THE FIREWALL
INQUIRING FOR REQUEST RESPONSES FROM THE RESPONSE QUEUE OR FROM THE SPECIAL DATABASE BY AN OUTPUT PROCESSING ELEMENT
71
72
RETURNING THE REQUEST RESPONSE TO THE CLIENT'S TERMINAL

FIG. 4

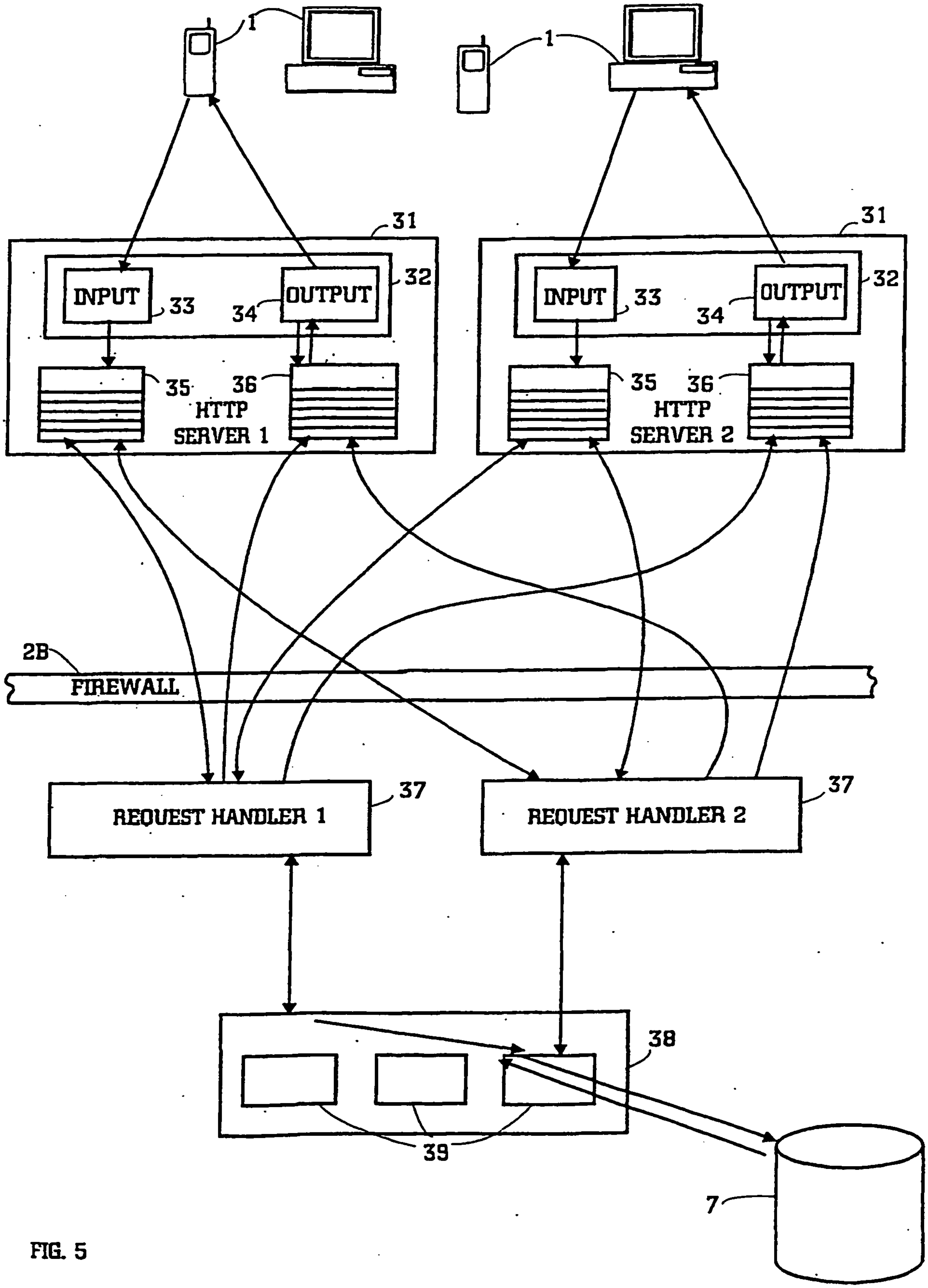

1
1
31
31
32
32
INPUT
33
OUTPUT
34
INPUT
33
OUTPUT
34
35
36
HTTP
SERVER 1
35
36
HTTP
SERVER 2
2B
FIREWALL
REQUEST HANDLER 1
37
REQUEST HANDLER 2
37
38
39
7

FIG. 5

INTERNATIONAL SEARCH REPORT

International application No.
PCT/FI 02/00280

A. CLASSIFICATION OF SUBJECT MATTER

IPC7: G06F 17/30, H04L 12/66, H04L 29/06

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

IPC7: G06F, H04L

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

SE,DK,FI,NO classes as above

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

EPO-INTERNAL, WPI DATA, PAJ

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y	US 6088796 A (CIANFROCCA, F. ET AL.), 11 July 2000 (11.07.00), claims 1-30, abstract	1-2,10-12,19
A		3-9,13-18,20
	--	
Y	US 6141759 A (BRADDY, R.G.), 31 October 2000 (31.10.00), column 6, line 11 - column 7, line 18, claims 1-33, abstract, cited in Application	1-2,10-12,19
	--	
P,Y	WO 0173522 A2 (NETFISH TECHNOLOGIES, INC), 4 October 2001 (04.10.01), claims 1-23, abstract	1-2,10-12,19
P,A		3-9,13-18,20
	--	

☒ Further documents are listed in the continuation of Box C. ☒ See patent family annex.

* Special categories of cited documents:

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier application or patent but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance: the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance: the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art

"&" document member of the same patent family

Date of the actual completion of the international search 13 August 2002	Date of mailing of the international search report 14 -08- 2002
Name and mailing address of the ISA/ Swedish Patent Office Box 5055, S-102 42 STOCKHOLM Facsimile No. +46 8 666 02 86	Authorized officer Roger Bou Faisal/LR Telephone No. +46 8 782 25 00

C (Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	US 5805803 A (BIRRELL, A.D. ET AL.), 8 Sept 1998 (08.09.98), column 2, line 20 - line 60, claims 1-19, abstract -- --------	1-20

INTERNATIONAL SEARCH REPORT

Information on patent family members

06/07/02

International application No.

PCT/FI 02/00280

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US 6088796 A	11/07/00	NONE	
US 6141759 A	31/10/00	US 6304967 B	16/10/01
WO 0173522 A2	04/10/01	AU 5384701 A	08/10/01
US 5805803 A	08/09/98	NONE	